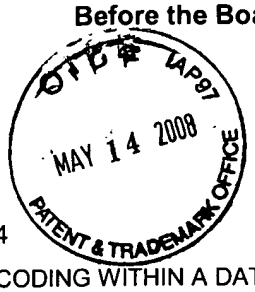
IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
Before the Board of Patent Appeals and Interferences

In re Patent Application of



Atty Dkt. JRL-550-509
C# M#
Confirmation No. 4228

SEAL, D. et al.

TC/A.U.: 2183

Serial No. 10/781,883

Examiner: Fennema

Filed: February 20, 2004

Date: May 14, 2008

Title: INSTRUCTION ENCODING WITHIN A DATA PROCESSING APPARATUS HAVING MULTIPLE INSTRUCTION SETS

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

☐ **Correspondence Address Indication Form Attached.**

☐ **NOTICE OF APPEAL**
Applicant hereby **appeals** to the Board of Patent Appeals and Interferences from the last decision of the Examiner twice/finally rejecting applicant's claim(s). $510.00 (1401)/$255.00 (2401) $

☒ An appeal **BRIEF** is attached in the pending appeal of the above-identified application $510.00 (1402)/$255.00 (2402) $ 510.00

☐ Credit for fees paid in prior appeal without decision on merits -$ ( )

☐ A reply brief is attached. **(no fee)**

☐ Petition is hereby made to extend the current due date so as to cover the filing date of this paper and attachment(s)
One Month Extension $120.00 (1251)/$60.00 (2251)
Two Month Extensions $460.00 (1252)/$230.00 (2252)
Three Month Extensions $1050.00 (1253/$525.00 (2253)
Four Month Extensions $1640.00 (1254/$820.00 (2254) $

☐ "Small entity" statement attached.

Less month extension previously paid on -$( )

TOTAL FEE ENCLOSED $ 510.00

☒ **CREDIT CARD PAYMENT FORM ATTACHED.**

Any future submission requiring an extension of time is hereby stated to include a petition for such time extension. The Commissioner is hereby authorized to charge any deficiency, or credit any overpayment, in the fee(s) filed, or asserted to be filed, or which should have been filed herewith (or with any paper hereafter filed in this application by this firm) to our **Account No. 14-1140.** A duplicate copy of this sheet is attached.

901 North Glebe Road, 11th Floor
Arlington, Virginia 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100
JRL:maa

NIXON & VANDERHYE P.C.
By Atty: John R. Lastova, Reg. No. 33,149

Signature: _____

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of

SEAL, D. et al. Atty. Ref.: 550-509

Serial No. 10/781,883 Group: 2183

Filed: February 20, 2004 Examiner: Fennema

For: INSTRUCTION ENCODING WITHIN A DATA PROCESSING APPARATUS HAVING MULTIPLE INSTRUCTION SETS

Before the Board of Patent Appeals and Interferences

BRIEF FOR APPELLANT

On Appeal From Final Rejection
From Group Art Unit 2183

John R. Lastova
NIXON & VANDERHYE P.C.
11th Floor, 901 North Glebe Road
Arlington, Virginia 22203-1808
(703) 816-4025
Attorney for Appellants
Seal et al. and
ARM Limited

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Patent Application of

SEAL, D. et al. Atty. Ref.: 550-509

Serial No. 10/781,883 Group: 2183

Filed: February 20, 2004 Examiner: Fennema

For: INSTRUCTION ENCODING WITHIN A DATA PROCESSING APPARATUS HAVING MULTIPLE INSTRUCTION SETS

* *

May 14, 2008

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

I. REAL PARTY IN INTEREST

The real party in interest is the assignee, ARM Limited, a United Kingdom corporation.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals related to this subject application. There are no interferences related to this subject application.

III. STATUS OF CLAIMS

Claims 1-33 are pending and rejected. Claims 1-6, 8-17, 19-28, and 30-33 are appealed.

IV. STATUS OF AMENDMENTS

An amendment was filed after final on January 7, 2008 and has been entered for purposes of appeal as indicated in the advisory action dated January 22, 2008.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The technology in this case relates to a data processing system having multiple instruction sets and the way in which such multiple instruction sets may be encoded. A data processing system supporting multiple instruction sets allows considerable flexibility in the way in which program operations may be represented, and that flexibility can yield improved code density. But the typical trade-off is increased hardware to support the multiple instruction sets. The inventors overcame this problem by arranging the encoding of the instruction sets such that a common subset of instructions has the same encoding, after variations due to storage order, e.g. endianness, have been compensated for. As a result, the data processing system implementation can be advantageously simplified by using a common decoding logic which reduces the hardware needed.

The non-limiting example embodiment described in the specification relates to unconditional coprocessor instructions that may be implemented using an ARM instruction set and equivalently using an enhanced Thumb© instruction set. Figure 4 shows the encoding for the ARM instruction set, and Figure 5 shows the encoding for enhanced Thumb© instruction set. Figure 6 shows how an example Thumb coprocessor instruction includes the same bytes as the ARM coprocessor instruction but in a different storage order. In this non-limiting example, the rules on page 12, lines 1-14 show how the correct instruction is sent to the instruction decoder 12 with the storage order compensated for.

The use of a common storage order compensated encoding for the unconditional coprocessor instructions in the ARM and enhanced Thumb instruction sets has considerable advantages for both forms of the instruction decoders 12, compared with the use of different encodings in each instruction set, in terms of reducing the amount of logic required and the amount of power consumed. A further advantage is that the coprocessor 22 only needs to be able to execute the same instructions as it could before the enhancement to the Thumb instruction set. As well as avoiding increases in the logic required by coprocessors and the power they consume, this also means that existing coprocessors do not need to be modified to be usable from the enhanced Thumb instruction set.

The following claim charts provide a mapping of the independent claims onto non-limiting example text from the specification and figures by reference

numerals were appropriate. This mapping is not intended to be used for claim construction.

1. (Previously Presented) An apparatus for processing data, said apparatus comprising:	Fig.1 and data processing apparatus 2. Page 6, line 6.
data processing logic configured to perform data processing operations; and	Processor core 3 and coprocessor 22 in Figure 1. Page 6, line 6-page 7, line 31.
an instruction decoder configured to decode program instructions specifying data processing operations to be performed by said data processing logic and to control said data processing logic to perform said data processing operations; wherein	Decoder 12 in Figure 1. Page 6, lines 8-9 and page 7, line 3-page 8, line 1 3.
said instruction decoder is operable in a first mode in which program instructions of a first instruction set are decoded and in a second mode in which program instructions of a second instruction set are decoded, a subset of program instructions of said first instruction set having a common bit-length and a common storage order compensated encoding with a subset of program instructions of said second instruction set such that, after compensating for storage order differences, all bits are identical and forming a common subset of instructions representing at least one class of instructions, said common subset of	Page 7, lines 32-33. Page 8, lines 3-12. E.g., unconditional coprocessor instructions and Figs. 4-6, page 11, lines 1-25 and page 13,

instructions controlling said data processing logic to perform the same data processing operations independent of whether said instruction decoder is operating in said first mode or said second mode.	line 31-page 14, line 12.

12. (Original) A method of processing data, said method comprising the steps of:	Page 4, lines 6-19. Fig.1 and data processing apparatus 2. Page 6, line 6.
performing data processing operations with data processing logic; and	Processor core 3 and coprocessor 22 in Figure 1. Page 6, line 7-page 7, line 31.
decoding with an instruction decoder program instructions specifying data processing operations to be performed by said data processing logic and controlling said data processing logic to perform said data processing operations; wherein	Decoder 12 in Figure 1. Page 6, lines 8-9 and page 7, line 3-page 8, line 1 3.
in a first mode program instructions of a first instruction set are decoded and in a second mode program instructions of a second instruction set are decoded, a subset of program instructions of said first instruction set having a common storage order compensated encoding with a subset of program instructions of said second instruction set and forming a common subset of instructions representing at least one class of instructions, said common subset of	Page 7, lines 32-33. Page 8, lines 3-12. E.g., unconditional

instructions controlling said data processing logic to perform the same data processing operations independent of whether said instruction decoder is operating in said first mode or said second mode.	coprocessor instructions and Figs. 4-6 and page 11, lines 1-25 and page 13, line 31-page 14, line 12.

23. (Previously Presented) A computer program product embodied in a storage medium for storing a computer program operable to control a data processing apparatus containing data processing logic operable to perform data processing operations, said computer program comprising:	Page 4, lines 21-34. Fig.1 and data processing apparatus 2. Processor core 3 and coprocessor 22 in Figure 1. Page 6, line 6-page 7, line 31.
program instructions of a first instruction set and program instructions of a second instruction set, that control said data processing logic to perform said data processing operations; wherein	Page 7, lines 32-33.
a subset of program instructions of said first instruction set have a common storage order compensated encoding with a subset of program instructions of said second instruction set and form a common subset of instructions representing at least one class of instructions, said common subset of instructions controlling data processing logic to perform the same data processing operations independent of whether instructions of said first instruction set or of said second instruction set are being decoded.	Page 8, lines 3-12. E.g., unconditional coprocessor instructions and Figs. 4-6 and page 11, lines 1-25 and page 13, line 31-page 14, line 12.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The first question to be decided by the Board is whether claims 1-6, 8-17, 19-28, and 30-33 are anticipated under 35 U.S.C. §102(b) based on Qureshi et al. (2004/00308856).

The second question to be decided by the Board is whether the rejection of claims 7, 18, and 29 under 35 U.S.C. §103 is improper.

VII. ARGUMENT

A. The Legal Standards for Anticipation

To establish that a claim is anticipated, the Examiner must point out where each and every limitation in the claim is found in a single prior art reference. *Scripps Clinic & Research Found. v. Genentec, Inc.*, 927 F.2d 1565 (Fed. Cir. 1991). Every limitation contained in the claims must be present in the reference, and if even one limitation is missing from the reference, then it does not anticipate the claim. *Kloster Speedsteel AB v. Crucible, Inc.*, 793 F.2d 1565 (Fed. Cir. 1986). Qureshi fails to satisfy this rigorous standard.

B. The Rejection Under 35 U.S.C. §102(b) Based on Qureshi Is Improper

Claims 1-6, 8-17, 19-28, and 30-33 stand rejected as being anticipated by Qureshi et al. (2004/00308856). This rejection is in error and should be reversed.

Qureshi is concerned with endianness. In a multi-byte memory block, "big endian" stores the most significant byte (MSB) in the lowest memory address, and "little endian" stores the least significant byte (LSB) in the lowest memory address. Qureshi allows for a system event register, present in a processor or other location of memory, to be accessed as either a big or little endian register depending on an operating system (OS) accessible bit for endian selection. The contents of the endian selection register are written during system boot up and are used by address decode logic to determine in what order to read/write/select the multiple bytes of an endian accessible memory location.

1. Qureshi Lacks The Claimed Instruction Decoder

In the final rejection, the Examiner maps the claimed "an instruction decoder configured to decode program instructions specifying data processing operations to be performed by said data processing logic and to control said data processing logic to perform said data processing operations," as recited in claim 1, to address decode logic in Qureshi. But this address decoder is not an instruction decoder that decodes program instructions. Paragraph [0005] of Qureshi states: "An address is sent to the address decode logic corresponding to an endian accessible memory block. The decode logic uses the selected address and the endian selection register to remap the bytes of the selected address location so that the MSB [most significant byte] and LSB [least significant byte] are provided in the order expected by the operating system." Thus, this decode logic in Qureshi

rearranges the order of the bytes of a selected address location. It does not decode program instructions. See also Qureshi's discussion of address decode logic 511 in paragraph [0018].

In the Advisory Action, the Examiner admits that Qureshi fails to teach this claimed feature: "Examiner will conceed [sic] that the Applicant is correct in this regard [that the address decode logic of Qureshi does not properly map onto the claimed limitation of a decoder configured to decode program instructions] and that the address decode logic that the Examiner mapped to does not decode program instructions." The Examiner "still asserts that there is a decode which does this in Qureshi, as the limitations of the claim require that the instruction decoder decode program instructions, and can be operable in both modes as disclosed. Therefore, if Qureshi does perform the rest of the claimed invention, Examiner believes it is a fair assertion to say that Qureshi has this decoder."

Although one might infer that Qureshi's CPU 401 in Figure 4 includes an instruction decoder of some sort, there is no evidence in Qureshi that such a decoder is what is claimed. The instruction decoder in claim 1 is "operable in a first mode in which program instructions of a first instruction set are decoded and in a second mode in which program instructions of a second instruction set are decoded." Qureshi does not describe a decoder or any other feature that operates in both claimed modes.

The point is that the claimed decoder is not disclosed in Qureshi, as the Examiner rightly admits. The only option left to the Examiner is to demonstrate that the claimed decoder is inherent in Qureshi. The Federal Circuit requires that "extrinsic evidence 'must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill'...'Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain thing may result from a given set of circumstances is not sufficient.'" *In re Robertson*, 49 USPQ2d 1949, 1950-51 (Fed. Cir. 1999). That extrinsic evidence is missing here.

2. Qureshi Lacks An Instruction Decoder Decoding Instructions From First and Second Instruction Sets

Qureshi fails to disclose "said instruction decoder is operable in a first mode in which program instructions of a first instruction set are decoded and in a second mode in which program instructions of a second instruction set are decoded," as recited in claim 1. The Examiner contends that the claimed first mode corresponds to the big endian mode in Qureshi, and the claimed second mode corresponds to the little endian mode. But storing data in one format or another format has nothing to do with decoder modes in which program instructions from a first instruction set and a second instruction set are decoded by the same decoder. In contrast, the big endian and little endian modes are simply different ways of storing data in a register. For example, in Tables 1 and 2 of Qureshi, the same piece of data ABCD059E is shown stored in big endian mode

and in little endian mode. Qureshi does not identify this data as a program instruction.

But even if the data value in Qureshi was said to be a program instruction, it is clear that it would be the same instruction whether it is stored in big endian or little endian mode. Because an instruction stored in the big endian mode is the same instruction as that stored in the little endian mode, the instruction stored in the big endian mode must be from the same instruction set as the instruction stored in the little endian mode. Thus, the two modes of Qureshi are not modes in which program instructions of a first instruction set and a second instruction set are decoded.

3. Qureshi Lacks The Claimed Subset Of Program Instructions

Qureshi also does not disclose "a subset of program instructions of said first instruction set having a common bit-length and a common storage order compensated encoding with a subset of program instructions of said second instruction set such that, after compensating for storage order differences, all bits are identical." Because Qureshi only describes rearranging the order of storage for a single piece of data, it does not disclose a subset of program instructions. Also, as shown above, a first instruction set and a second instruction set are not disclosed in Qureshi. So this third claim feature also cannot be disclosed by Qureshi.

In the second paragraph of the Advisory Action, the Examiner argues "that the two subsets claimed are not necessarily different subsets, because unless the two are claimed as distinct, there is no requirement to read them as the same subset." Although this sentence is difficult to understand, the Examiner seems to be suggesting that the first and second subsets are the same subset. First, the Examiner's suggestion is an admission that Qureshi does not teach the claimed first and second instruction sets or the first and second respective subsets of these instruction sets. Second, the Examiner is not entitled to adopt an unreasonable interpretation of the claims. The plain meaning of the terms "first" and "second" is to distinguish between multiple, distinct features. A person of ordinary skill in the art would understand that the terms "first instruction set" and "second instruction set" refer to two distinct instruction sets; it is unreasonable and contrived to say otherwise. Third, the Examiner's interpretation must also be consistent with the meaning of the claim terms as understood in light of the specification. The specification clearly contradicts the Examiner's interpretation.

4. Qureshi Lacks The "Common Subset Of Instructions" Claim Element

Given that the claimed instruction decoder, first and second modes, and first and second instruction sets are not disclosed by Qureshi, Qureshi also lacks "forming a common subset of instructions representing at least one class of instructions, said common subset of instructions controlling said data processing logic to perform the same data processing operations independent of whether said

instruction decoder is operating in said first mode or said second mode." Qureshi only describes rearranging the order of storage for a single piece of data. That is simply not the same thing as forming a common subset of program instructions.

In the fourth paragraph of the Advisory Action, the Examiner states that "in the claimed invention, when saying that you have two instruction sets, and when compensating for their storage differences, are identical in every way, and perform the exact same operation." But this mischaracterizes what is claimed. In the independent claims, a subset of program instructions of the first instruction set has a common storage order compensated encoding with a subset of program instructions of the second instruction set. Thus, the claimed first and second subsets of instructions, from their respective instruction sets, are such that after compensating for storage differences, all bits are identical. Qureshi does not describe any subset of program instructions—let alone the claimed first and second subsets. Again, storing a data value in two different storage format is not the same as instructions from two different instruction sets.

The Examiner argues in the third paragraph of the advisory action that "a subset is a term which can encompass all, none, or part of a set, thus in Qureshi's case, the subset of instructions... is the entire set." But if every instruction in the first and second instruction sets was part of the common subset of program instructions, then the first and second instruction sets would be the same instruction set. If all the instructions are from the same instruction set, then there

is only one instruction set. This "only one instruction set" interpretation renders the claim language meaningless, and thus, cannot be the correct interpretation. There would be no need for an instruction decoder that is "operable in a first mode in which program instructions of a first instruction set are decoded and in a second mode in which program instructions of a second instruction set are decoded" if all of the all the instructions are from the same instruction set. Thus, Qureshi does not disclose the claimed common subset of instructions formed by a subset of program instructions from the first instruction set and the subset of program instructions from the second instruction set.

5. The Dependent Claim Rejections Under 35 U.S.C. 102 Are Improper

Regarding claims 2, 13, and 24, Qureshi lacks an "instruction decoder [that] is operable to use common portions of said data processing logic to execute instructions of said common subset of instructions." The Examiner has not been able to point to an instruction decoder disclosed in Qureshi. The decode logic identified by the Examiner in [0005] is address decode logic and not instruction decode logic. There is no teaching of using "one set of logic" to decode instructions from two different instruction sets.

Claims 3, 14, and 25 recite that "said common subset of instructions includes a class of instructions being coprocessor instructions operable to control coprocessor data processing operations using coprocessor logic common to said

first instruction set and said second instruction set." Coprocessor instructions are not disclosed in Qureshi.

Claims 4, 15, and 26 recite that "all unconditional coprocessor instructions are within said common subset." Given that coprocessor instructions are not disclosed in Qureshi, this additional feature is also missing.

Regarding claims 8, 19, and 30, the Examiner wrongly attempts to map different storage formats of a data value to operating an instruction decoder in the two different claimed modes to decode instructions from different instruction sets.

The Examiner's rationale for claims 9, 10, 11, 20, 21, 22, 31, 32, and 33 simply does not address the specific features relating to "at least one program instruction generating different result data values includes a program counter [or status register] value as an input operand." These claim features are not found in Qureshi, and the Examiner's reasoning about reading data out of order is simply not relevant to a common subset of instructions formed from two different instruction sets.

D. The Rejection of Claims 7, 18, and 29 under 35 U.S.C. §103 Is Improper

The Examiner proposes adding McFarland's teachings of an x86 processor that can execute variable length instructions to Qureshi's system for accessing a system event register as either a big or little endian register depending on an operating system (OS) accessible bit for endian selection. But there is no reason

to make this combination. The Examiner assumes that this combination "would make variable-length instructions one of the instruction sets Qureshi deals with." But this statement assumes a fact that is not present—that Qureshi teaches dealing with multiple instruction sets. The Examiner must be confusing Qureshi with the claimed technology, because it is the claimed technology (and not Qureshi) that "deals with" multiple instruction sets, and it is claims 7, 18, and 29 that deal with one of those instruction sets including variable length instructions.

VIII. CONCLUSION

As demonstrated above, Qureshi lacks multiple claim features, and as a result, the anticipation rejection is in error. The obviousness rejection is based on improper hindsight. The Board should reverse all of the rejections.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: 

John R. Lastova
Reg. No. 33,149

JRL/maa
Appendix A - Claims on Appeal

IX. CLAIMS APPENDIX

1. (Previously Presented) An apparatus for processing data, said apparatus comprising:

data processing logic configured to perform data processing operations; and

an instruction decoder configured to decode program instructions specifying data processing operations to be performed by said data processing logic and to control said data processing logic to perform said data processing operations; wherein

said instruction decoder is operable in a first mode in which program instructions of a first instruction set are decoded and in a second mode in which program instructions of a second instruction set are decoded, a subset of program instructions of said first instruction set having a common bit-length and a common storage order compensated encoding with a subset of program instructions of said second instruction set such that, after compensating for storage order differences, all bits are identical and forming a common subset of instructions representing at least one class of instructions, said common subset of instructions controlling said data processing logic to perform the same data processing operations independent of whether said instruction decoder is operating in said first mode or said second mode.

2. (Previously Presented) The apparatus as claimed in claim 1, wherein said instruction decoder is operable to use common portions of said data processing logic to execute instructions of said common subset of instructions.

3. (Previously Presented) The apparatus as claimed in claim 1, wherein said common subset of instructions includes a class of instructions being coprocessor instructions operable to control coprocessor data processing operations using coprocessor logic common to said first instruction set and said second instruction set.

4. (Previously Presented) The apparatus as claimed in claim 3, wherein all unconditional coprocessor instructions are within said common subset.

5. (Previously Presented) The apparatus as claimed in claim 1, wherein said first instruction set is a fixed length instruction set of N-bit instructions.

6. (Previously Presented) The apparatus as claimed in claim 5, wherein N is one of 32 or 16.

7. (Previously Presented) The apparatus as claimed in claim 1, wherein said second instruction set is a variable length instruction set.

8. (Previously Presented) The apparatus as claimed in claim 1, wherein at least one program instruction within said common subset of instructions performs common data processing operations in either said first mode or said second mode but

generates different result data values depending upon whether said instruction decoder is operating in said first mode or said second mode.

9. (Previously Presented) The apparatus as claimed in claim 8, wherein said at least one program instruction generating different result data values includes a program counter value as an input operand.

10. (Previously Presented) The apparatus as claimed in claim 9, wherein a different relationship is maintained between said program counter value and an address of an instruction being executed depending upon whether said instruction decoder is operating in said first mode or said second mode.

11. (Previously Presented) The apparatus as claimed in claim 8, wherein said at least one program instruction generating different result data values includes a program status register value as an input operand.

12. (Original) A method of processing data, said method comprising the steps of:

performing data processing operations with data processing logic; and

decoding with an instruction decoder program instructions specifying data processing operations to be performed by said data processing logic and controlling said data processing logic to perform said data processing operations; wherein

in a first mode program instructions of a first instruction set are decoded and in a second mode program instructions of a second instruction set are decoded, a subset of program instructions of said first instruction set having a common storage order compensated encoding with a subset of program instructions of said second instruction set and forming a common subset of instructions representing at least one class of instructions, said common subset of instructions controlling said data processing logic to perform the same data processing operations independent of whether said instruction decoder is operating in said first mode or said second mode.

13. (Previously Presented) The method as claimed in claim 12, wherein common portions of said data processing logic are used to execute instructions of said common subset of instructions.

14. (Previously Presented) The method as claimed in claim 12, wherein said common subset of instructions includes a class of instructions being coprocessor instructions operable to control coprocessor data processing operations using coprocessor logic common to said first instruction set and said second instruction set.

15. (Previously Presented) The method as claimed in claim 14, wherein all unconditional coprocessor instructions are within said common subset.

16. (Previously Presented) The method as claimed in claim 12, wherein said first instruction set is a fixed length instruction set of N-bit instructions.

17. (Previously Presented) The method as claimed in claim 16, wherein N is one of 32 or 16.

18. (Previously Presented) The method as claimed in claim 12, wherein said second instruction set is a variable length instruction set.

19. (Previously Presented) The method as claimed in claim 12, wherein at least one program instruction within said common subset of instructions performs common data processing operations in either said first mode or said second mode but generates different result data values depending upon whether said instruction decoder is operating in said first mode or said second mode.

20. (Previously Presented) The method as claimed in claim 19, wherein said at least one program instruction generating different result data values includes a program counter value as an input operand.

21. (Previously Presented) The method as claimed in claim 20, wherein a different relationship is maintained between said program counter value and an address of

an instruction being executed depending upon whether said instruction decoder is operating in said first mode or said second mode.

22. (Previously Presented) The method as claimed in claim 19, wherein said at least one program instruction generating different result data values includes a program status register value as an input operand.

23. (Previously Presented) A computer program product embodied in a storage medium for storing a computer program operable to control a data processing apparatus containing data processing logic operable to perform data processing operations, said computer program comprising:

program instructions of a first instruction set and program instructions of a second instruction set, that control said data processing logic to perform said data processing operations; wherein

a subset of program instructions of said first instruction set have a common storage order compensated encoding with a subset of program instructions of said second instruction set and form a common subset of instructions representing at least one class of instructions, said common subset of instructions controlling data processing logic to perform the same data processing operations independent of whether instructions of said first instruction set or of said second instruction set are being decoded.

24. (Previously Presented) The computer program product as claimed in claim 23, wherein common portions of said data processing logic are used to execute instructions of said common subset of instructions.

25. (Previously Presented) The computer program product as claimed in claim 23, wherein said common subset of instructions includes a class of instructions being coprocessor instructions operable to control coprocessor data processing operations using coprocessor logic common to said first instruction set and said second instruction set.

26. (Previously Presented) The computer program product as claimed in claim 25, wherein all unconditional coprocessor instructions are within said common subset.

27. (Previously Presented) The computer program product as claimed in claim 23, wherein said first instruction set is a fixed length instruction set of N-bit instructions.

28. (Previously presented) The computer program product as claimed in claim 27, wherein N is one of 32 or 16.

29. (Previously presented) The computer program product as claimed in claim 23, wherein said second instruction set is a variable length instruction set.

30. (Previously presented) The computer program product as claimed in claim 23, wherein at least one program instruction within said common subset of instructions performs common data processing operations when instructions of either said first instruction set or said second instruction set are being decoded but generates different result data values.

31. (Previously presented) The computer program product as claimed in claim 30, wherein said at least one program instruction generating different result data values includes a program counter value as an input operand.

32. (Previously presented) The computer program product as claimed in claim 31, wherein a different relationship is maintained between said program counter value and an address of an instruction being executed depending upon whether said instruction decoder is operating in said first mode or said second mode.

33. (Previously presented) The computer program product as claimed in claim 30, wherein said at least one program instruction generating different result data values includes a program status register value as an input operand.

X. EVIDENCE APPENDIX

There is no evidence appendix.

XI. RELATED PROCEEDINGS APPENDIX

There is no related proceedings appendix.